Department of Computer Science
Texas State University

CS4337 Project Report

Coauthor
Daniel Almanza
ggs46@txstate.edu

Coauthor
Johnathan Carrizales
ryo8@txstate.edu

Abstract

This project focuses on developing a computer vision system capable of automatically detecting and classifying playing cards in a live demonstration. We implemented a single stage YOLO based detection/classification model, and a two stage YOLO detection and CNN classification model trained on a Kaggle dataset containing 2,757 labeled card images spanning 53 distinct classes (joker included).

Focusing on the single stage pipeline, the annotations were converted into YOLO format and trained as a YOLOv8-n model for 10 epochs on CPU. The final Model achieved 94.3% accuracy, 0.95 precision, and 0.942 recall when evaluated per image. Additionally, implemented a real time pipeline that runs at ~30 fps and can detect and classify playing cards from a webcam. The system performs well under stable conditions but is limited by its ability to detect although once detected makes accurate classifications.

For the two-stage pipeline, the annotations were similarly converted to YOLO format and trained using YOLOv8-n model for 10 epochs on the CPU. After that, a CNN classification was added for classification. The final model has 99% accuracy, 99% precision, and 99% recall for all the ranks. For the suits it had a 100% accuracy, 100% precision, and 100% recall. It was always stable and had good detection but bad classification.

Together, these two approaches demonstrate different strategies for solving the same problem. One model performing detection end-to-end, and the other dividing the task it detection and classification, the results highlight the strengths and limitation of each method and show how deep learning techniques can be applies effectively to structed visual recognition tasks.

1 Introduction

The task in this project is to build a computer vision system capable of recognizing playing cards in real time trained on images. This requires both the location of a card and the specified class label (rank and suit). Although the problem seems simple at first, playing card recognition introduces several practical challenges. Cards can vary in appearance, lighting variance, camera angle, motion blur, partial occlusion, and background clutter. Additionally, the key identifying features of a card are small high-detail regions that require strong detection and classification performance.

To address this problem, we implemented two approaches. A single stage detection and classification YOLO model, and a two stage YOLO detection and CNN classification model. Our goals were to

- Build a complete end to end card recognition pipeline.
- Compare single-stage versus two-stage approaches.

- Evaluate accuracy, precision, recall, and confusion matrices,
- Deploy a working, real-time demonstration.

2 Methods

The project was implemented using two complementary pipelines. Each pipeline contributes different strengths to the overall objective of robust playing card recognition. The methodology below describes the design, training, and evaluation of the single stage YOLO detector, followed by a dedicated section for the two-stage approach.

Single-Stage YOLO Card Detection and Classification

The single-stage pipeline uses a YOLOv8 model to detect a playing card and classify it directly into one of 53 card classes. This approach treats the problem as a combined localization + classification task.

Dataset Preparation

- A curated dataset of 2,757 labeled images was used, each containing a single card annotated with its bounding box and class label.
- Labels were converted from VOC XML to YOLO format, producing files in the format: class x_center y_center width hegith
- Automated integrity checks verifies
  - Consistent labeling across 53 classes
  - Valid coordinate ranges
  - One label per image
  - Matching image label file pairs

Preprocessing and Augmentation:

To improve robustness and compensate for variation in lighting, angle, and distance:

- All images were resized to 640 x 640 during training
- YOLO's built-in augmentations were enabled, including random flips, HSV jitter, and mosaic composition.
- Particular attention was given to preserving fine corner details, which are critical for distinguishing rank and suit.

Model Training

Training was performed using YOLOv8n, chosen for its CPU efficiency. The key training setting were:

- 10 epochs, batch size 4.
- AdamW optimizer.
- Mosaic augmentation enabled for early epochs.
- Confidence threshold tuned for small-symbol detection.

Across training, losses steadily decreased, and evaluation of metrics indicated strong generalization across the dataset.

Evaluation

Model performance was evaluated using:

- Mean average precision
- Precision, recall, and F1-score per class
- A full 53 x 53 confusion matrix for misclassification analysis

Final performance reached:

- 71.5% mAP@50
- 94.3% per image classification accuracy
- High precision/recall across the majority of classes

Real Time Deployment

A real time inference system was implemented using OpenCV:

- The YOLO detector processes webcam frames in real time.
- Bounding boxes and predicated labels are drawn on screen

This deployment demonstrated consistent performance when the card is fully visible and stable within frame.

Two-Stage YOLO Card Detection and Classification

The two-stage pipeline uses a YOLOv8 model to detect a playing card and a CNN clasifier to classify it directly into one of 52 card classes. This approach treats the problem as two seperate problems one to detect a card and the other to classify it.

Dataset Preparation

- A curated dataset of 2,757 labeled images was used, each containing a single card annotated with its bounding box and class label.
- Labels were converted from VOC XML to YOLO format, producing files in the format: class x_center y_center width hegith
- Automated integrity checks verifies
    - Consistent labeling across 53 classes
    - Valid coordinate ranges
    - One label per image
    - Matching image label file pairs
- For the two stage the cards were cropped and organized into two class folders one for rank and the other for suits

Preprocessing and Augmentation:

To improve robustness and compensate for variation in lighting, angle, and distance:

- All images were resized to 640 x 640 during training
- YOLO's built-in augmentations were enabled, including random flips, HSV jitter, and mosaic composition.
- Cropped card images for the CNN were normalized and resized to the classifier input size.
- Special care was given to preserving corner features, since small rank/suit symbols determine class identity.

Stage 1 YOLOv8 Detection Training:

YOLOv8n was selected for its fast inference speed and strong performance.
Key training settings:

- 10 epochs, batch size 32
- AdamW optimizer
- Mosaic augmentation early in training
- Tuned confidence threshold for small-feature detection

Stage 2 CNN Detection Training:

The second stage uses a CNN trained on cropped card images to classify the card into **53 possible ranks/suits**.

- Input: cropped YOLO card region
- Output: 53-class prediction
- Architecture: lightweight convolutional network for fast inference
- Loss: cross-entropy
- Augmentations: rotation, brightness shift, and slight perspective jitter

This stage focuses on fine-grained features that YOLO does not specialize in.

Evaluation:

Model performance was evaluated using:

- Precision, recall, and F1-score per class
- Two full 4x4 and 52x52 confusion matrix for misclassification analysis

Final performance reached:

- 99% per image classification accuracy
- High precision/recall across the majority of classes

Real-Time Deployment:

The final system runs in real time using OpenCV:

1. YOLOv8 detects the card in each webcam frame.
2. The detected region is cropped and passed to the CNN classifier.
3. Bounding boxes and predicted labels are drawn on-screen around the full card.

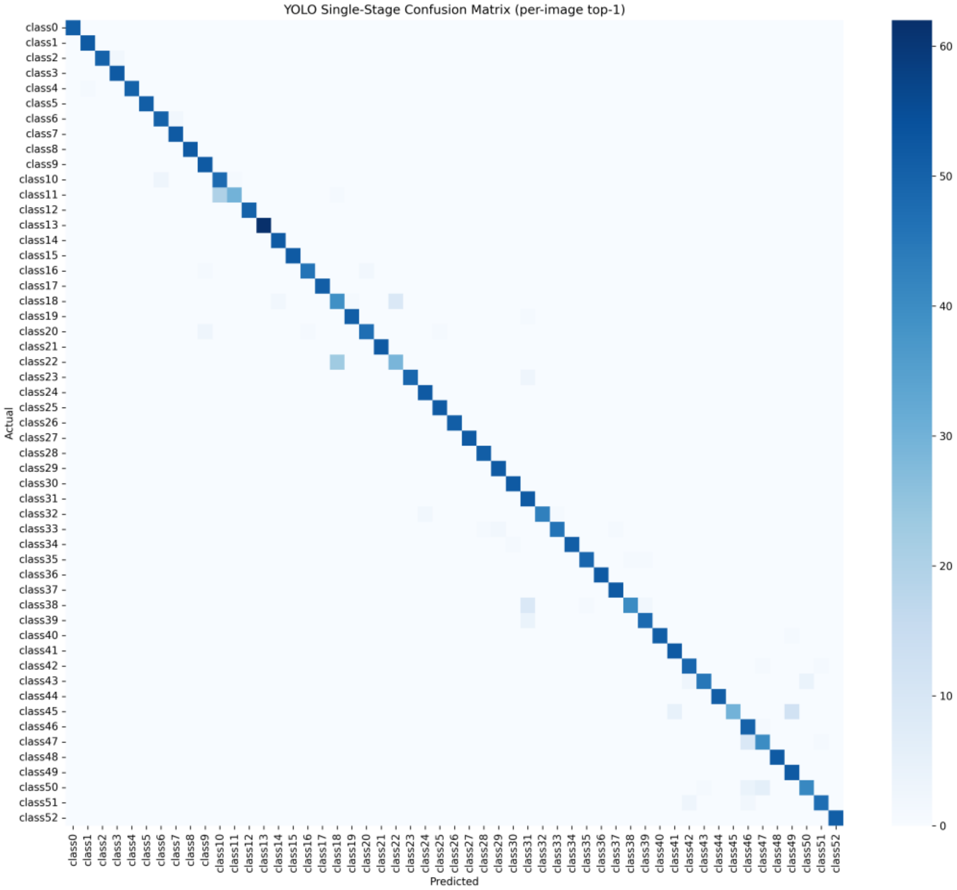
The two-stage design performed consistently when the card was fully visible and steady in view.

3 Results

This section presents the experimental setup, quantitative outcomes, qualitative observations, and failure case analysis for both pipelines.

Metric	**Value**
mAP@50	0.715
MAP@ 50-95	0.447
Overall Precision	0.948
Overall Recall	0.943
Top-1 classification accuracy	94.3%

These Metrics indicate that the model is highly reliable at identifying the correct card class when a clear bounding box is produced. Performance decreases when evaluating quality at stricter IoU thresholds, which is expected for small-object datasets.

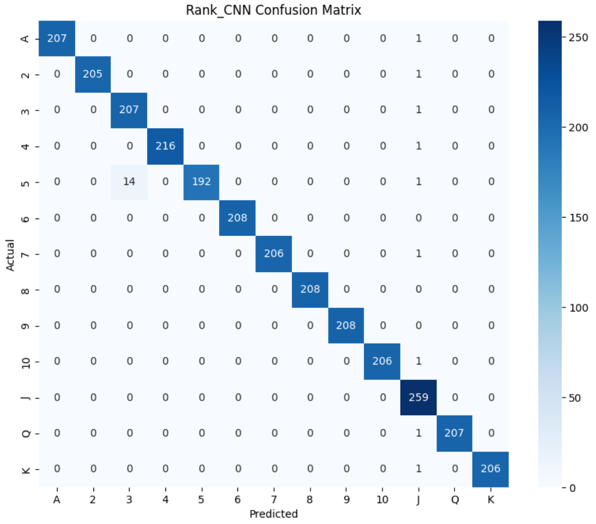


Above is the generated confusion matrix where we can see, most predictions lie perfectly along the diagonal indicating string accuracy across the dataset, misclassifications occur mainly between similar ranks and suits. This also shows that the model overtrained, and could be an indicator as to why during the demo the model could classify cards accurately, but only once it was successfuly detected.

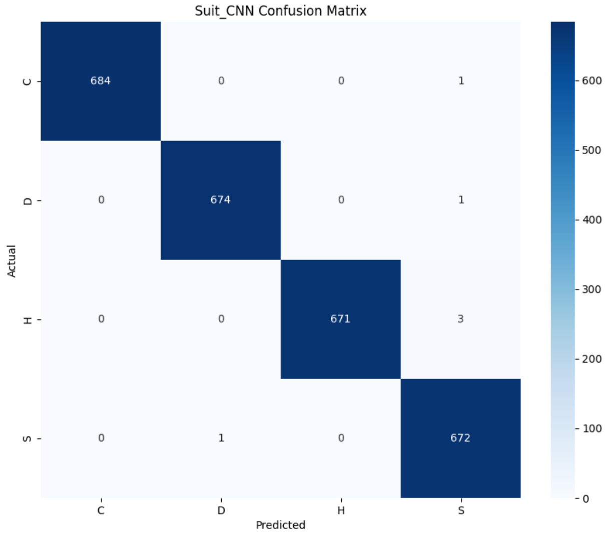
This is the two-stage pipeline for both rank and suit

Metric	**Value**
Overall Precision- rank	0.99%
Overall Recall –rank	0.99%
Top-1 classification accuracy – rank	99%
Overall Precision- suit	100%
Overall Recall –suit	100%
Top-1 classification accuracy – suit	100%

Two-stage pipeline Confusion Matrix



The confusion matrix above for the ranks is correctly predicting the cards all be it with some being detected wrong like the jacks prediction having 1s above and below it



The suits confusion matrix shows that it correctly detected the correct suit but with the diamonds having 1 miss prediction and the spades having several miss predictions.

4 Discussion

The results from both approaches show that playing card recognition is highly feasible, but each method handles the problem's challenges differently. The single-stage YOLO model demonstrated strong quantitative performance, achieving over 94% top-1 accuracy and performing reliably on static, well-framed images. When detections were stable, the model classified cards with high confidence and correctly distinguished most ranks and suits.

However, real-world testing revealed several limitations to single-stage detection. Because cards occupy very small regions within webcam frames, the model often produces tiny bounding boxes around corner symbols rather than detecting the entire card. This behavior reduced classification stability and made the system sensitive to distance, motion, and viewing angle. Small-object detection, motion blur, and low-resolution card features all contributed to missed or unstable predictions.

In contrast, the two-stage pipeline where detection and classification are handled separately showed better control over the recognition process. By isolating cropped card regions and applying specialized CNN classifiers for rank and suit, the two-stage method mitigated some of the small-object challenges faced by YOLO. Fine-grained classification improved because the dedicated CNNs received consistent, close-up inputs rather than full-frame detections.

Across both systems, common failure cases included visually similar cards (e.g., hearts vs. diamonds, or numerically close ranks) and scenarios with poor lighting or rapid movement. These issues highlight the difficulty of fine-grained classification and the need for training data that reflects real deployment conditions.

Overall, the project shows that single-stage exceled at classification and simplicity, while the two-stage pipeline offers stronger control over detection quality. Combining ideas from both such as improving full-card detection, enhancing data augmentation, or using higher-resolution inputs would likely yield even more robust results in future work.

5 Conclusion

This project demonstrated that automated playing card recognition can be effectively achieved using computer vision techniques, with both single-stage and two-stage approaches offering meaningful strengths. The single-stage YOLO model proved fast, simple to deploy, and capable of high accuracy on clean images, while the two-stage pipeline provided more stable fine-grained classification through its dedicated rank and suit networks. Together, these results highlight the trade-off between speed and precision when designing real-time recognition systems.

The experiments also revealed important challenges. Detecting small cards in real-world webcam footage remains difficult, particularly under motion, distance, or lighting variations. Both pipelines occasionally struggled with visually similar cards, and the single-stage detector was especially sensitive to small-object localization. These limitations point toward clear opportunities for improvement, such as using higher-resolution inputs, stronger augmentations, or refining the detection stage to better capture full-card regions.

Overall, the project successfully delivered a working card recognition system, a comparison of two fundamentally different architectures, and a deeper understanding of how model design impacts performance in real deployment scenarios. Future work could explore hybrid methods, larger datasets, or more advanced architectures to further strengthen robustness and real-time accuracy.

References

Shah, Jay Pradip. “The Complete Playing Card Dataset.” Kaggle, 2021, [Online]. Available: https://www.kaggle.com/datasets/jaypradipshah/the-complete-playing-card-dataset